STAT515 Project1

Yiqun Hu

03/25/2020

1. build **simCorn** function

```
knitr::opts_chunk$set(error = TRUE)
```

```
simCorn <- function(overallEffect=0, fertilizerEffect=c(0,0,0),
                    rowEffect=c(0,0,0), colEffect=c(0,0,0),
                    seed=set.seed(NULL), dist=rnorm(n=9,...), ...){

  Fertilizer = as.factor(fertilizerEffect)
  Row = as.factor(rowEffect)
  Column = as.factor(colEffect)

  #Error check
  if (length(overallEffect) != 1 | !is.numeric(overallEffect))
    stop("Error, overrallEffect is not a numeric or length 1!")


  if (length(fertilizerEffect) ==3  && is.numeric(fertilizerEffect))
    {
    fertilizerEffect <- fertilizerEffect
  }else{
    stop("Error, fertilizerEffect is not a numeric or length 3!")
  }

  if (length(rowEffect) ==3 && is.numeric(rowEffect)) {
    rowEffect <- rowEffect
  }else{
    stop("Error, rowEffect is not a numeric or length 3!")
  }

   if (length(colEffect) ==3 && is.numeric(colEffect)){
     colEffect <- colEffect
   }else {
    stop("Error, colEffect is not a numeric or length 3!")
   }

  if (is.null(seed)) {
  #set.seed()
  } else if (is.vector(seed)){
  set.seed(seed)
  } else {
    stop("Error, seed is neither an integer or NULL!!")
```

```
  }

  #Equation1
  Yield = overallEffect + fertilizerEffect + rowEffect +
    colEffect + dist

  #return data frame
  return(data.frame(Fertilizer , Row , Column, Yield,
                    stringsAsFactors = TRUE))

}
```

Check simCorn() function with three examples

```
#Example1
y <- simCorn()
y
```

```
##   Fertilizer Row Column       Yield
## 1          0   0      0  0.08796924
## 2          0   0      0 -0.67691124
## 3          0   0      0  0.15118633
## 4          0   0      0 -0.22213890
## 5          0   0      0  1.26140507
## 6          0   0      0  0.83238901
## 7          0   0      0  0.56524516
## 8          0   0      0 -0.22300929
## 9          0   0      0 -0.16642741
```

```
#Example2
y <- simCorn(overallEffect = 10, seed = 2123,
             dist = rgamma(n=9, shape = 2))
y
```

```
##   Fertilizer Row Column    Yield
## 1          0   0      0 14.83727
## 2          0   0      0 10.45424
## 3          0   0      0 13.13900
## 4          0   0      0 10.47095
## 5          0   0      0 10.90779
## 6          0   0      0 13.44940
## 7          0   0      0 10.77832
## 8          0   0      0 11.49251
## 9          0   0      0 10.62710
```

```
#Example3
mu <- 7
alpha <- c(1,2,3)
beta <- c(2,2,1)
gamma <- c(3,3,2)
y <- simCorn(overallEffect = mu, fertilizerEffect = alpha,
             rowEffect = beta, colEffect = gamma, seed = 29429,
             dist =rnorm(n=9,mean = 3, sd = 2))
y
```

```
##   Fertilizer Row Column    Yield
```

```
## 1          1   2      3 19.97551
## 2          2   2      3 16.08501
## 3          3   1      2 16.82301
## 4          1   2      3 13.00537
## 5          2   2      3 16.31535
## 6          3   1      2 13.89126
## 7          1   2      3 15.24435
## 8          2   2      3 16.98722
## 9          3   1      2 19.10382
```

2. Help narriative
simCorn

Description
simulate data frame of observed Yields for three fertilizers A, B, C.

Usage
simCorn <- function(overallEffect=0, fertilizerEffect=c(0,0,0), rowEffect=c(0,0,0), colEffect=c(0,0,0), seed=set.seed(NULL), dist=rnorm(n=9,...), ...)

Arguments
overallEffect specified overall mean yield. Defaults to 0.
fertilizerEffect specified fertilizer effect. Defaults to c(0,0,0)
rowEffect specified row effect. Defaults to c(0,0,0)
rowEffect specified column effect. Defaults to c(0,0,0)
seed an option random seed. Defaults to NULL.
dist a function name that generates a random sample
from a distribution in R. Defaults to rnorm.
... addtional arguments.

Details

The value returned by the function would be a data frame that contains the simulated data. The column names in the data frame match those given above (**Fertilizer, Row, Column,** and **Yield**). The **Fertilizer, Row,** and **Column** columns should be factors while **Yield** should be numeric.

If **seed** is **NULL** then the seed of the random number generator should not be set. If **seed** is equal to an integer then the **set.seed** function should be executed using the specified **seed**. If **seed** is neither an integer or NULL then an error should be returned.

dist is the name of an R function that simulates random samples. You may assume that the first argument of this function is the sample size, and that all of the remaining arguments of the function will be supplied by the user though the special argument ....

The function would do error checking to insure that the user supplied arguments are of the correct type and shape. Errors would be reported to the user with detailed error messages.

The data be simulated using $Y = \mu + \alpha_k + \beta_l + \gamma_m + \epsilon_{klm}$

Value
Appropriate the result can be a data frame (which is a special type of list).

For the "lm" method, the result is a data frame with an attribute "seed". If argument seed is NULL, the attribute is the value of .Random.seed before the simulation was started; otherwise it is the value of the argument with a specified attribute with value set.seed(seed).

Examples

#Example1
y <- simCorn(overallEffect = 10, seed = 2123,
dist = rgamma(n=9, shape = 2))

#Example2
mu <- 7
alpha <- c(1,2,3)
beta <- c(2,2,1)
gamma <- c(3,3,2)
y <- simCorn(overallEffect = mu, fertilizerEffect = alpha,
rowEffect = beta, colEffect = gamma, seed = 29429,
dist =rnorm(n=9,mean = 3, sd = 2))

3.

```
#simulation1
pValues1 <- replicate(100,{
  y <- simCorn(overallEffect = 10, fertilizerEffect = c(0,0,0),
               rowEffect = c(0,0,0), colEffect = c(0,0,0) ,
               dist = rnorm(n=9, mean = 0, sd = 1))
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data = y)
anova(fitCorn)$"Pr(>F)"[1]
})
```

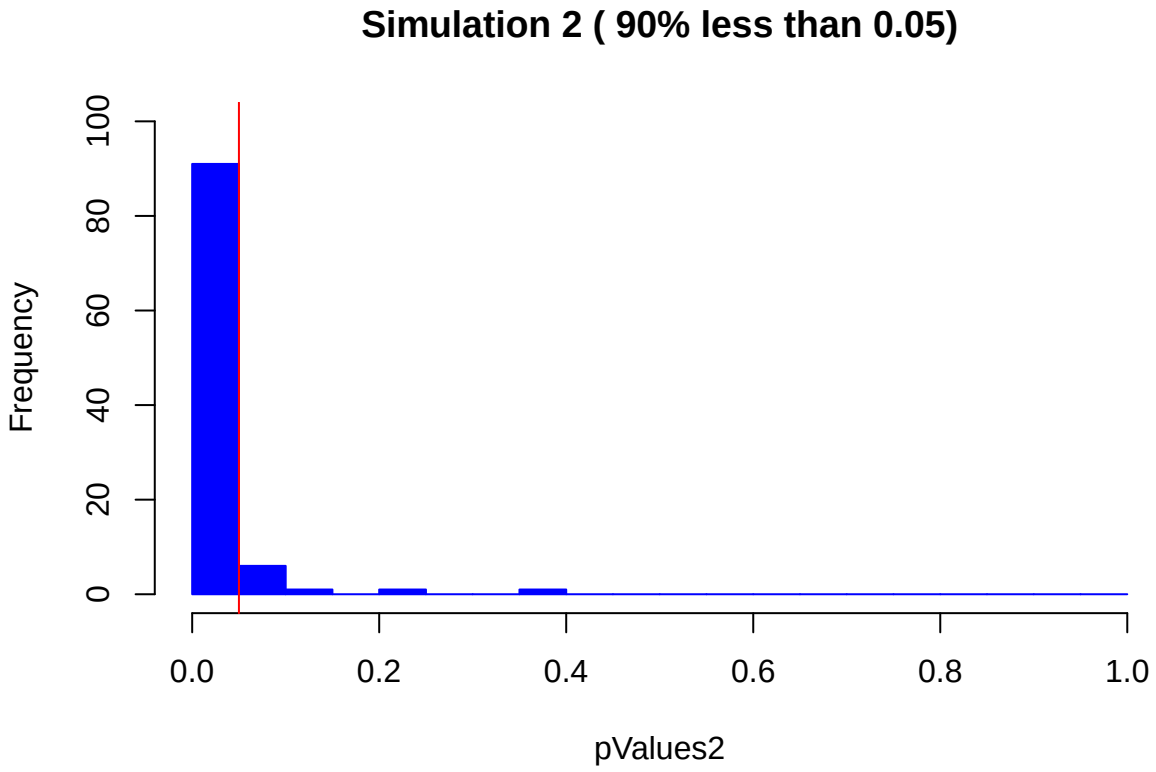
```
## Error in `contrasts<-`(`*tmp*`, value = contr.funs[1 + isOF[nn]]): contrasts can be applied only to
```

```
{hist(pValues1,breaks = seq(0.00,1.00,0.05),col = "blue",
     border = "blue",main = "Simulation 1 ( % less than 0.05)", ylim = range(seq(0,100,5)))
abline(v=.05, col ="red")}
```

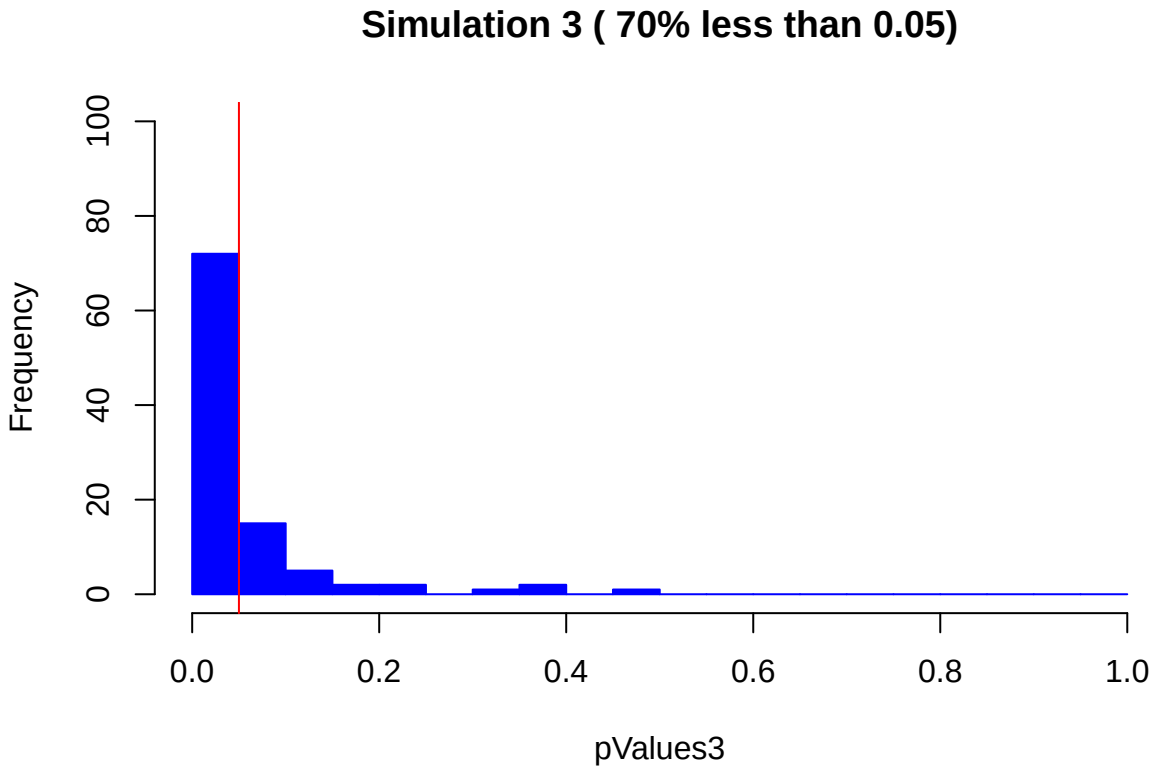
```
## Error in hist(pValues1, breaks = seq(0, 1, 0.05), col = "blue", border = "blue", : object 'pValues1'
```

Simulation1 does not exist!

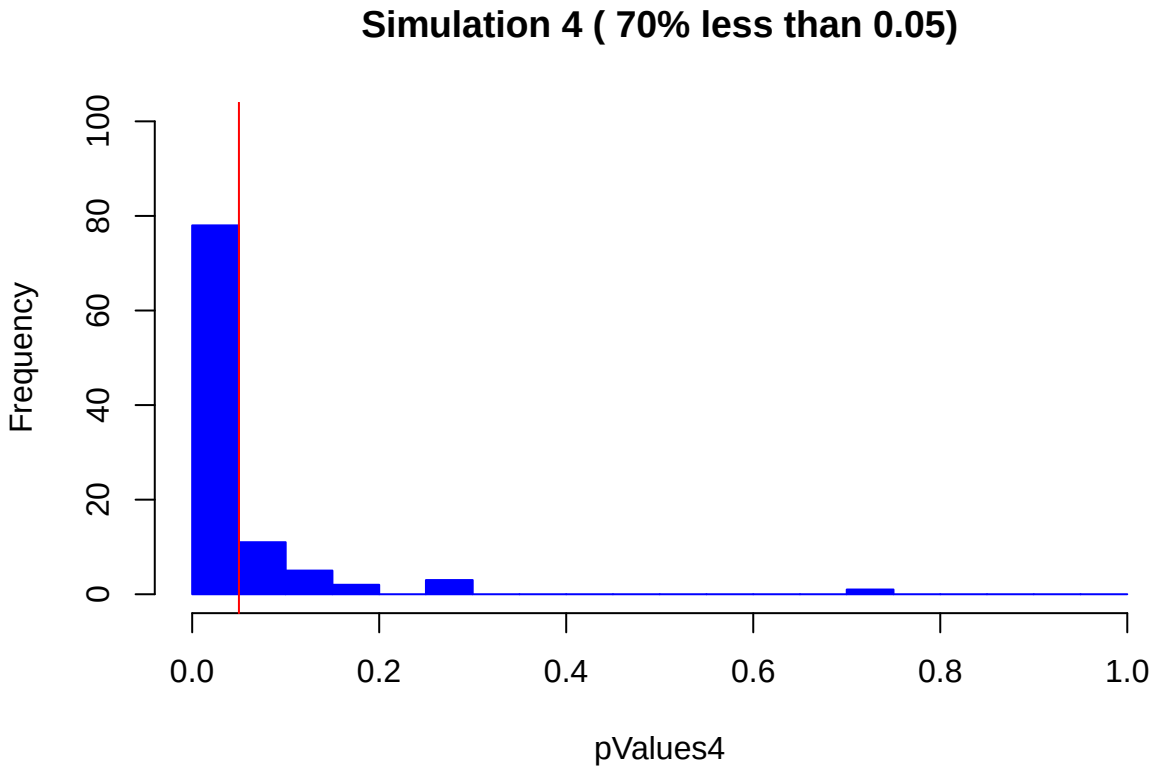
```
#simulation2
pValues2 <- replicate(100,{
  y <- simCorn(overallEffect = 10, fertilizerEffect = c(1,2,3),
               rowEffect = c(0,0,1), colEffect = c(0,0,1) ,
               dist = rnorm(n=9, mean = 0, sd = 1))
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data = y)
anova(fitCorn)$"Pr(>F)"[1]
})
{hist(pValues2,breaks = seq(0.00,1.00,0.05),col = "blue",
     border = "blue",main = "Simulation 2 ( 90% less than 0.05)", ylim = range(seq(0,100,5)))
abline(v=.05, col ="red")}
```



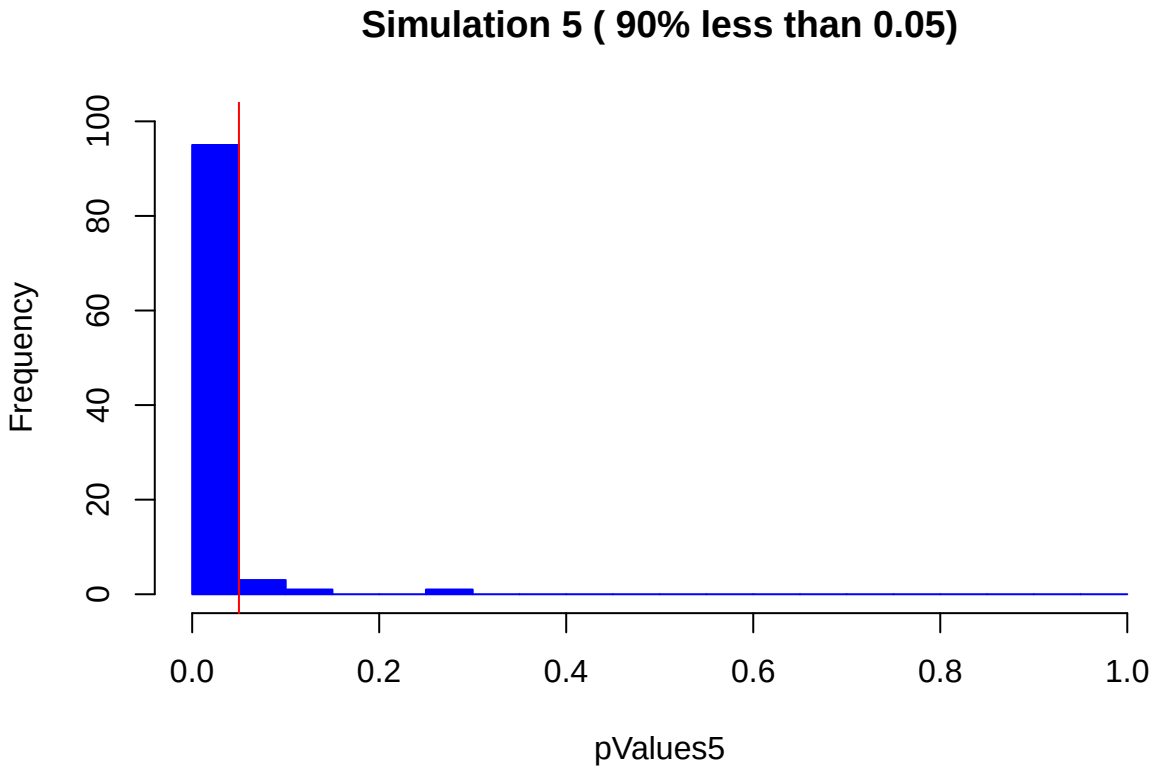

```
#simulation3
pValues3 <- replicate(100,{
  y <- simCorn(overallEffect = 10, fertilizerEffect = c(1,2,3),
               rowEffect = c(1,0,1), colEffect = c(0,1,1) ,
               dist = rnorm(n=9, mean = 0, sd = 1))
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data = y)
anova(fitCorn)$"Pr(>F)"[1]
})
{hist(pValues3,breaks = seq(0.00,1.00,0.05),col = "blue",
     border = "blue",main = "Simulation 3 ( 70% less than 0.05)", ylim = range(seq(0,100,5)))
abline(v=.05, col ="red")}
```



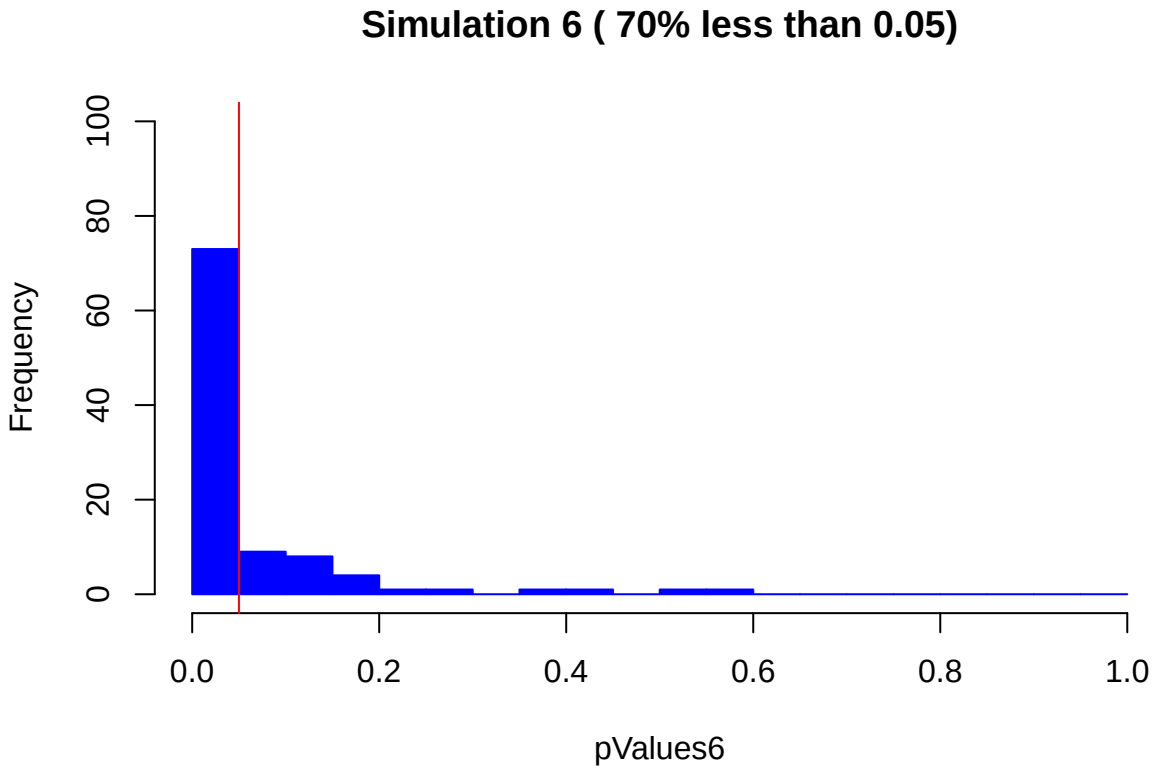

```
#simulation4
pValues4 <- replicate(100,{
  y <- simCorn(overallEffect = 10, fertilizerEffect = c(1,2,3),
               rowEffect = c(0,1,0), colEffect = c(0,1,0) ,
               dist = rnorm(n=9, mean = 0, sd = 1))
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data = y)
anova(fitCorn)$"Pr(>F)"[1]
})
{hist(pValues4,breaks = seq(0.00,1.00,0.05),col = "blue",
     border = "blue",main = "Simulation 4 ( 70% less than 0.05)", ylim = range(seq(0,100,5)))
abline(v=.05, col ="red")}
```



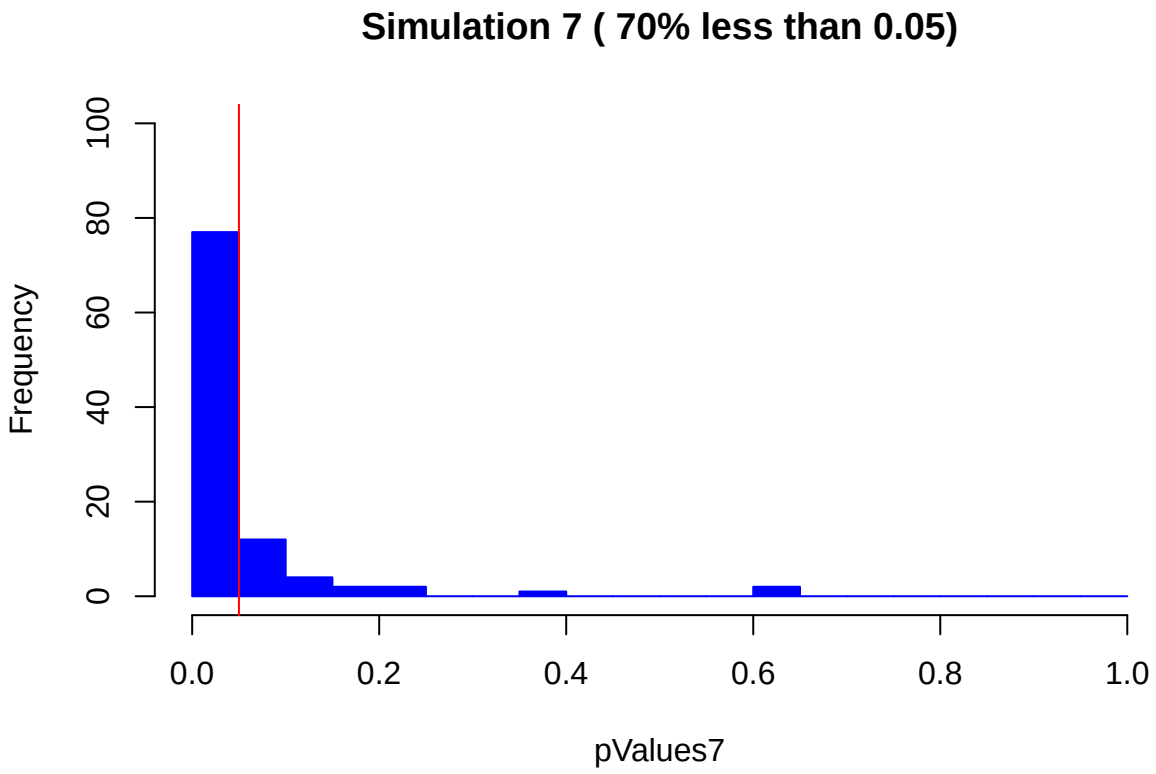

```
#simulation5
pValues5 <- replicate(100,{
  y <- simCorn(overallEffect = 10, fertilizerEffect = c(1,2,3),
               rowEffect = c(0,0,1), colEffect = c(0,0,1) ,
               dist = rexp(n=9, rate = 1))
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data = y)
anova(fitCorn)$"Pr(>F)"[1]
})
{hist(pValues5,breaks = seq(0.00,1.00,0.05),col = "blue",
     border = "blue",main = "Simulation 5 ( 90% less than 0.05)", ylim = range(seq(0,100,5)))
abline(v=.05, col ="red")}
```




```
#simulation6
pValues6 <- replicate(100,{
  y <- simCorn(overallEffect = 10, fertilizerEffect = c(1,2,3),
               rowEffect = c(1,0,1), colEffect = c(0,1,1) ,
               dist = rexp(n=9, rate = 1))
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data = y)
anova(fitCorn)$"Pr(>F)"[1]
})
{hist(pValues6,breaks = seq(0.00,1.00,0.05),col = "blue",
     border = "blue",main = "Simulation 6 ( 70% less than 0.05)", ylim = range(seq(0,100,5)))
abline(v=.05, col ="red")}
```




```
#simulation7
pValues7 <- replicate(100,{
  y <- simCorn(overallEffect = 10, fertilizerEffect = c(1,2,3),
               rowEffect = c(0,1,0), colEffect = c(0,1,0) ,
               dist = rexp(n=9, rate = 1))
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data = y)
anova(fitCorn)$"Pr(>F)"[1]
})
{hist(pValues7,breaks = seq(0.00,1.00,0.05),col = "blue",
     border = "blue",main = "Simulation 7 ( 70% less than 0.05)", ylim = range(seq(0,100,5)))
abline(v=.05, col ="red")}
```


Simulation 7 (70% less than 0.05)
Frequency
0
20
40
60
80
100
0.0
0.2
0.4
0.6
0.8
1.0
pValues7